Identification and Authentication when Users have Multiple Accounts

W.R.Shockley
Cyberscape Computer Services
8051 Vierra Meadows Place
Salinas, CA 93907

Abstract

Most security models assume that each user has only a single account. This simplifies the enforcement of a security policy by allowing rules about individuals to be replaced by rules about accounts. However, the assumption fails badly for large-scale networks, because no realistic approach exists for ensuring that it is true. It is therefore preferable to acknowledge that users may have multiple accounts and make adjustments to the identification and authentication mechanisms. Examples of security policies where the potential for multiple accounts makes a difference are given. A simple mechanism for account alias detection is described that supports the correct enforcement of these policies event when account aliases may exist. The key idea is to separate the authentication function—determining that the owner of the account is present—from the identification function determining whether the owners of two different accounts are the same individual. Identification is a natural application for biometric technology. The use of biometrics for identification alone has significant operational and cost benefits over its use for authentication. A system that used conventional authentication techniques coupled with biometric identification would seem to be optimal.

1 Terminology

Before an individual can successfully use a shared computer system to perform user-level work, typically an account must be established for that individual. This operation, which will be called user registration, is a sequence of steps performed by an administrator who will be called a registrar. User registration is a security-critical operation, and the registrar is trusted to conform to procedural and administrative controls ensuring that individuals not authorized to own accounts are not registered, and that any account information pertaining to a newly-registered individual is properly verified and entered into the account database.

The account itself is represented by a protected, security-critical data record of some sort that contains these data. The set of account records is keyed by some kind of account identifier that uniquely identifies individual account records. Often "user friendly" substitute keys, such as a user name, are supported that can also be used to identify the account record.

Among the data generated during the registration operation are data that can be used later to authenticate the new user when that user establishes a session on the system. In general, we can think of the authentication data as consisting of two parts: an authenticator which is held by the actual user, and an authenticand which is stored as part of the account data record. An authentication algorithm is built into the trusted computing base that, given an ⟨authenticator, authenticand⟩ pair determines, with an appropriately high probability of success, whether or not the given authenticator and authenticand match. We think of the registrar as generating (or causing to be generated) an initial ⟨authenticator, authenticand⟩ pair for each new account, with the authenticator being provided to the owner of the new account and the authenticand being stored in the account record for later use. It is often the case (e.g., when the authenticator is a password) that the procedure for generating and distributing a new⟨ authenticator, authenticand⟩ pair can be automated so that already registered users may be permitted to replace their own authentication data without further intervention by a registrar.

This simple model of an authentication technology, consisting of

- a method for generating a non-repeating sequence of ⟨authenticator, authenticand⟩ pairs

- a method for determining whether a given authenticator was generated in association with with a given authenticand
- a procedure to be followed by a registrar when assigning new accounts to users, and
- an optional method for replacing a registered user's old ⟨authenticator, authenticand⟩ with a new one

is sufficiently abstract to cover a wide range of existing authentication approaches, such as password or passphrase-based authentication, the use of smart cards or other physical tokens for authentication, the use of biometric information, techniques based on symmetric key encryption as found in Kerberos [MIL87], or on asymmetric key encryption as proposed for DSSA [GAS89,LIN90].

A typical scenario for using an authentication technology to limit the access to a computer system to authorized users is the following:

1. The Prospective user first claims to own a particular account by supplying its account identifier or its surrogate, a user name,
2. A trusted component of the system fetches the nominated account record and prompts the user for an authenticator.
3. The user supplies an authenticator.
4. The trusted log-on procedure uses the comparison method to determine whether this authenticator together with the authenticand found in the nominated account record are valid authenticator, authenticand pair, granting or denying access to the computer system depending on the result

Of course this procedure is vulnerable to situations where the authenticator has been compromised, either deliberately by the owner of the account or registrar, or from some other cause. Some procedural means of recovering from an authenticator compromise, e.g., generation of a new ⟨authenticator, authenticand⟩ pair, is therefore often considered a requirement.

We will define the identification function as the determination, for two given accounts, whether or not these accounts are owned by the same individual. A third result—"no determination"—is also admitted. While this may not be an immediately intuitive definition of what the word "identification" means, it turns out to be the most useful one for dealing with the problems described later in this paper.

In order to support a distinct identification function, we assume that an account record may optionally include identification data collected by a registrar from the owner of the account when the account was created. In principle, the identification data should have the following properties:

- It should uniquely identify the individual. i.e., no two individuals should have the same identification data
- It should be difficult for a given individual to forge or produce false identification data whether at different times or indifferent places.

While weaker kinds of identification data may be imagined, the use of a biometric identification technology comes immediately to mind. These technologies generally have the following functional components:

- a biometric reader can be used to capture physical data from an individual—e.g. fingerprint, retina scan, voiceprint.
- a reduction algorithm is available for reducing the raw biometric data to a canonical representation we will call a biometric profile.
- a profile comparison algorithm that, given a pair of biometric profiles tells, with some appropriately small probability of error, whether or not the profiles represent measurements taken from the same individual.

2 Authentication and Identification Technologies Contrasted

The thesis of this paper is that while the requirements driving the choice of an authentication technology and those driving the choice of an identification technology are similar, they are also in certain environments (e.g., the Internet) and for certain policies incompatible—which motivates us to carefully separate the two functions so that we can use the best technologies for each. In particular, the requirement on an authentication technology to be able to recover expeditiously from an authenticator compromise clashes with the requirement on an identification technology that it must be difficult to change an individual's identification data.

As a concrete example, suppose that a biometric technology is chosen for the authentication function (the usual security application heretofore suggested for these technologies). The ⟨ authenticator, authenticand ⟩ pair is taken as:

- for the authenticand, a biometric profile captured from the user during the registration operation
- for the authenticator, a biometric profile captured from the user during log-on

The biometric comparison algorithm is used as the authentication matching algorithm. This approach has a fatal flaw if recovery from authenticator compromise is an issue (and it usually is). By definition, it is difficult to change a user's biometric profile (otherwise, the biometric technology simply wouldn't work as advertised). Therefore, if a user's biometric profile is stolen (i.e., its bits are known to a penetrator) one has the worry forever after that the penetrator can successfully bypass a log-on biometric reader somewhere and use those bits to spoof the log-on software.

An example of the converse class of problems might involve the use of public-key authentication technology in environments where identification (i.e., detection of alias accounts) is an issue. Suppose, for example, that (as is asserted by the DSSA designers) a user's public key is taken as the user's "real" identity. In environments where it is impossible to prevent a given user from acquiring multiple registrations (e.g., Internet) it is quite possible for a user to accumulate several distinct public key "identities"—i.e., by being registered by different registrars. As we will see, this becomes a problem for certain security policies.

3 The Single Account Assumption

In my experience, most security modeling efforts have assumed, either tacitly or explicitly, that no user has more than a single account. This is an attractive simplifying assumption because it allows rules about individuals (e.g., security policies) to be immediately re-expressed as a set of homologous rules about accounts that can then be enforced by a properly designed reference monitor.

For example, if we examine a conventional access control policy expressed in terms of access control lists (ACLs), what we will find stored in the ACLs are typically account identifiers or surrogates for account identifiers (i.e., user names). The access of individuals to a protected object is really controlled indirectly: the ability of a user to obtain a terminal process is controlled by authentication, making sure that the individual owns the account used to label the process. When the process makes a request to access a protected object, typically it is the user name associated with the process that is matched against the user name found in the ACL.

More recent systems addressing the problems of authentication and account maintenance systems in networks, e.g., DSSA, use references to a different substitute key—viz. the account's current authenticand (public key). This allows the account record to be renamed (i.e., moved around as part of the name service object hierarchy). However, since authenticands are generated on an account-by-account basis there is still no guarantee that multiple accounts for the same user (which may have been installed by completely different registrars) are associated with the same authenticand.

Neither style of system precludes situations where an individual owns multiple accounts—nor does enough information exit to definitively prevent or detect such a condition. A few security policies where this makes a difference will be described later. It has been all-to-common practice to simply assert as an implied or explicit axiom for a security policy model that a given individual possesses at most a single account.

What is wrong with this assumption?

- In practice, even for small shared systems or networks, the rule is often honored more in the breach than the observance. One commonly finds that, for very practical reasons, operators may have both "system" and "user-level" accounts; "group accounts" are common, and if different machines are networked, various users may have distinct accounts on various machines.
- When it is left up to a system operator to enforce the "one user, one account" rule administratively, the enforcement becomes administratively difficult as the system grows to incorporate several operators and more users than re personally known to all of them—yet no help is provided in determining when multiple accounts exist.
- For still larger networks, such as Internet, it becomes unrealistic to even suppose that such a rule could be enforced as users are registered. Either the owner of a new account would have to be trusted to provide a list of all other accounts owned by that individual, or some service would have to be provided that searched the network for all accounts owned by the user being registered.

The first is manifestly insecure, and the second highly impractical.

- Finally, for some administrative domains a policy of "one user, one account" may be considered inconsistent with privacy regulations. It can be argued that the notion of "privacy" includes the right to perform at least some activities in such a way that they cannot be correlated with other activities performed by the same individual—e.g., that it must be possible to perform some functions anonymously.

4 Multiple Account Model and Mechanism

It appear necessary, then, to embrace the requirement to permit users to own multiple accounts. A basic model incorporating such a requirement is neither complex nor surprising in content. We permit users to own multiple accounts. Two accounts owned by the same user are called alias accounts. It is not generally the case that alias accounts were created by the same registrar, nor do we require a registrar to determine, when a user opens a new account, whether an alias exists. As for the conventional model, part of the registration procedure includes the generation of a valid ⟨authenticator, authenticand⟩ pair with the authenticator being given to the account owner, and the authenticand stored within the account record.

Some accounts, when they are opened, will include within their account record additional identification data, collected from the registrar from the user at the time the account is opened. If a biometric technology is used to support identification, this means the user must be present at the time the account is opened so that a biometric profile can be computed. Weaker forms of identification technology (e.g., Social Security number, mother's maiden name, etc.) might not require the physical presence of the user but would result in a correspondingly weaker identification system. Note that identification data is not intended to be used during user authentication but for identification only in support of specific policies. We will see later why it is desirable to make this exclusion.

Accounts that contain the optional identification data are called identified accounts. Accounts without identification data are called anonymous accounts. It is assumed that the record structure for account records allow identified accounts to be distinguished from anonymous accounts. In practice, type information identifying the particular authentication and identification technologies employed would be included as well.

When a user logs in to initiate a session, the trusted computing base authenticates the user just as for the Single Account Model. The identification data is not used by the authentication system in any way. As usual, the result of a successful authentication is that a terminal process is created for the user. We have thus established after a successful authentication that the user, on whose behalf the process is executing, is the owner of the account associated with the process.

The identification data, copied from the account record, is also associated with the process as part of the process security data Since we know after authentication that the current user owns the account, we know that the identification data is also the data for the current user even though it was not collected at the time of log-on but at the time of registration. Of course, we assume throughout that the account record has been properly protected by the TCB.

Now, whenever the process requests access to an object, the identification data associated with the process is available for potential use by the reference validation mechanism. In a distributed environment, it would be included and protected with other security context data as part of the request context for remote accesses.

It remains to describe how the identification data might be used by the reference validation mechanism to enforce specific security policies: the discussion above is focused on how the proper identification data is supplied to the RVM.

5 Selected Security Policies that Need Identification Data

The existence of alias accounts is of little consequence to the enforcement of many traditional security policies. For example, policies such as the DoD mandatory policy for the classification of label do not depend the identity of individuals, but only their clearances. It makes no difference from which of many properly registered accounts an individual might choose to access information with respect to this policy. Similarly, a policy that grants access to an object based on account identifier or user name is not compromised should the user happen to own a different account: the user would be able to access the object in question from one account, but (inconveniently but securely) not from the other. If the user should try. to access the object from the wrong account, the access would simply fail.

However, there are fairly interesting policies whose enforcement fails in the presence of alias accounts unless identification data supporting detection of alias accounts is provided as part of the request context.

5.1 Explicit Denial

One such policy is part of the DoD Criteria for Trusted Systems [DOD85] for the higher ratings—viz., the requirement to support the explicit denial of access. The intent of this requirement is that it should be possible to positively deny access to a given protected object by a specific individual.

If a conventional implementation is used (i.e., using ACLs containing account names or equivalents) and alias accounts exist, the policy cannot be accurately enforced. Denying access to at particular account does not ensure that the accounts' owner cannot gain access using alias account. As has been noted earlier, a search for all possible alias accounts at the time the explicit denial is entered or encountered is unfeasible in large networks.

The mechanism described earlier solves the problem easily. An explicit denial as in, e.g., an ACL, is constrained to contain a reference to an identified account that is treated as a reference to the identification data contained in the named account record. When a request to access the protected object is mediated, the RVM follows this reference to obtain the identification data (i.e., a biometric profile) from the referenced and uses the biometric comparison algorithm to match it against the profile contained as identification data in the request context The first data refers to the individual intended by whoever set up the ACL: the second to that individual on whose behalf the request is being made. Of course, all requests made from an anonymous account must be denied.

An unfeasible search is avoided, because all of the information needed to make the decision can be directly located. Note that it is quite possible that the profile located via the ACL, and the profile contained in the request context may well come from different identified accounts. The policy is enforced because if a denied individual makes a request from any identified account, the profile will match and the request therefore rejected, while if from an anonymous account (or one using a different identification technology) the request will be denied because no profile of the right type is provided with the request.

5.2 Separation of Duties

Another class of policies that are very important for many production-level applications are policies related to the separation of duties [CLA87]. Generally, these policies require that specific steps of given business process must be performed by distinct individuals. The intent of such policies is generally combined one of reducing error rates and inhibiting fraud by requiring collusion among would-be perpetrators. Approaches for enforcing a separation of duties policy involving the use of special-purpose accounts, groups, or role-based mechanisms fail badly in the presence of alias anonymous accounts because the possibility exists that a lone malicious individual could perform the critical steps from different accounts (e.g., on perhaps the day he or she changed jobs).

Such policies are very naturally specified and enforced where identification data, over and above authentication data, is available. When the first step is performed, the security context (including identification data) is captured and stored in a protected location (i.e., within the security perimeter). When a request for initiation of the second step is received, in addition to whatever other security checks may be indicated, the identification data from the second request is compared to that stored for the first. If there is a match, the requests presumably come from the same individual and initiation of the second step is blocked.

Again, since identification data must be used, requests to perform either step from anonymous accounts must be rejected.

5.3 Support for Privacy

In an earlier section, reference was made to policies for the privacy of information and activities. Such policies are considered of more or less importance than security policies depending upon the administration involved.

The author considers one aspect of "privacy" (as something distinct from "security") as the ability of an individual to perform some activities (e.g., private activities such as managing one's checkbook) that cannot be definitively correlated with other activities performed by the same individual (e.g., public or official activities for which an individual is held accountable). Put another way, individual accountability for public or official actions must be possible, but without implying disclosure of all of an individual's activities, however personal.

The provision in the model for anonymous accounts is intended to capture this possibility. Presumably, objects for which public accountability is required would be protected by ACLs prohibiting access from any anonymous account, while individuals could freely enjoy the use of the system via anonymous accounts for unofficial or personal business.

This approach would also carry over into the audit system. Identification data, when part of the request context, would be captured as part of the audit trail allowing the actions of a user from different identified accounts to be correlated.

Some have questioned the desirability of anonymous accounts should technology supporting identified accounts become widely available. This question is part of the ongoing discussion regarding the trade-off between hampering the investigative capabilities of law enforcement agencies and providing support for individual privacy. This is clearly a social issue: some administrative domains might choose to support a policy of "no anonymous accounts" so that all of an individual's activities within the domain might be, in principle, subject to correlation, while others might choose to permit registration of anonymous accounts by some or all of the user community. However, this is a social, not a technical issue. I have included the notion of an anonymous account so that the model could be applied to a wide range of social policies.It is worth noting that under the definitions given, most user accounts today providing access to on-line services are anonymous in the technical sense. (Information such as "user name" is provided by the user,—not verified by the registrar.) Nothing at all prevents a given user from purchasing as many accounts as desired, under whatever pseudonyms or address desired—as long as the monthly bill is paid! A social decision to prohibit such accounts in the future thus represents a distinct erosion of privacy.

5.4 As a Primary Access Control Mechanism

Finally it can be noted that the use of identification data in place of account identifier as a primary tool for granting individual access to data merits consideration. One could envision an "access control rule language" that extended current account names with a modifier indicating that it is the identification data that is to be compared, not the account identifier, in order to permit access. For example, a rule such as

$$grant(read, write, execute) to FOO \qquad (1)$$

would be interpreted to mean "permit access only if from account 'FOO' " while

$$grant(read, write, execute) to!FOO \qquad (2)$$

(where $F00'$ is an identified account) would be interpreted to mean "permit access to individual owning account FOO from any identified account owned by that individual". The benefit intended by this suggestion is strictly operational—as anyone who has gotten involved with trying to access "private" objects from multiple accounts will know. It is painful to try to set up ACLs in such a way that such objects can routinely be accessed from all of your accounts!

6 Implications

In the environment where support for alias accounts is important—viz, large-scale networks, the issue arises as to how the identification data is to be protected. However, it is always piggy-backed on an already protected entity—viz., security-critical account records from the account registry, security-critical process contexts, or security-critical request contexts. The identification data is protected in exactly the same way as the authenticand, using whatever mechanism is used to protect the authenticand – or large networks, typically end-to-end cryptographic sealing to make the record involved tamper-detectable.

Secondly, it should be noted that the authentication subsystem proper makes no use of the identification data. This means that inclusion of identification data in the security-critical records that carry it cannot disturb the correctness of the authentication system.

In fact, one can go further. Since authentication does not depend in anyway on the secrecy of the identification data, there is no compromise if identification data is disclosed. This result may seem at first startling. What must be understood is that it is the association between identification data and the account record that must be protected within the security perimeter. A malicious user or user process has no way of forging such an association, providing the underlying protection mechanism is sound, even if it knows what the identification data is. Authenticators must be protected from disclosure—identification data does not. What we are trusting, in the final analysis, is that the account registrar properly collected the real identification data from the owner of new account for inclusion in the account record.

Supposing that the security perimeter is broached and the account record compromised, recovery after repair of the perimeter is straightforward. A new authenticator and authenticand must be generated for the compromised accounts. However, if a cryptographically sealed record containing the biometric profile can be located and validated anywhere in the system the biometric profile therein may be safely re-used—only if no such record can be located must the users biometrics be re-read.

The proposed use of biometric technology is roughly an order of magnitude cheaper that the usually suggested use as an authentication mechanism. If it is used for authentication, then a relatively expensive biometric reader must be located at every log-in point. If the use of the biometric profile is restricted to identification alone, readers are needed only at designated registration workstations as no profile needs to be measured at the time of log-on.

It is difficult, however, to see how requiring the physical presence of the user at the time of an initial registration can be avoided.

7 Acknowledgements

The basic mechanism described in this paper, restricted to the use of biometric technology and cryptographically sealed account records, was invented jointly by the author and George Gajnak while employees of Digital Equipment Corporation. Mr. Gajnak deserves equal credit for the Multiple Account Model, while any mistakes in this paper are my own. We have jointly filed for patent protection internationally for the use of the mechanism to enforce all policies described in this paper. All rights to the invention have been assigned to Digital Equipment Corporation. The contents of this paper and disclosure of this take invention should not be construed as or product commitment by Digital Equipment Corporation.

Acknowledgement is also due to the anonymous referees of this paper for their comments and clarifications.

Annotated Bibliography

Surprisingly little seems to appear in the literature concerning identification and authentication policies although specific authentication technologies have received widespread attention.

[CAR88] S. Carlton, J. Taylor, and J. Wyzynski, "Alternative Authentication Techniques," *Proceedings, 11th National Computer Security Conference*, Baltimore, MD, Oct. 1988 pp. 333–338. Discusses general characteristics of biometric technologies for use for authentication but does not consider the impact of compromise. No distinction is made between the authentication and identification functions.

[CLA 87] D. Clark and D. Wilson, "A Comparison of Commercial and Military Security Policies," *Proceedings of the 1987 Symposium on Security and Privacy*, Oakland, CA, May 1987, pp.233–248. Contains much basic material on separation of duties policies.

[DOD85] Department of Defense, DoD Trusted Computer System Evaluation Criteria, DoD 5200.28-STD, Washington, D.C. 1985 [U.S.Government Printing Office 008-000-00461-7]. As well as defining requirements for explicit denial, the sections on control objectives and background rationale contains basic information about authentication, identification, and audit requirements for the Department of Defense.

[AS88] M.Gasser, *Building a Secure Computer System*, Van Nostrand Reinhold Company, New York NY.,1988. Everything that is said about the distinction between identification and authentication on pp. 20–22 is correct (if non-specific) but no practical application of the distinction is elaborated.

[GAS89] M. Gasser, A. Goldstein, C. Kaufmann, B.Lampson, "The Digital Distributed System Security Architecture," *Proceedings, 12th National Computer Security Conference*, Baltimore, MD, Oct. 1989, pp 305–319. Description of the DSSA public-key based security and authentication framework for networked systems. No specific support for the identification function is described.

[LIN90] John Linn, "Practical Authentication for Distributed Computing," *Proceedings of the 1990 IEEE Symposium on Security and Privacy*, IEEE Computer Society, 1988, pp 31–39 presents a practical and scalable public-key authentication design (an instance of DSSA described in the previous paper). Such a design could be easily enhanced to support the additional identification function.

[MIL87] S.P. Miller, R. C. Neumann, J.I. Schiller, and J.H. Saltzer (Massachusetts Institute of Technology), Project Athena Technical Plan Section E.2.1, "Kerberos Authentication and Authorization System" describes a symmetric-key based authentication system for networks that threatens to become an ad hoc standard.